Proyecto Final Q-Learning

Carla Domenech

Aprendizaje Automático No Supervisado Y Por Refuerzo

Universidad San Jorge

Fecha: 26/05/2025

Profesor: Sergio Gracia Borobia

INDICE

INTRODUCCIÓN

El aprendizaje por refuerzo es una rama del aprendizaje automático donde un agente aprende a tomar decisiones interactuando con un entorno, con el objetivo de obtener la mayor recompensa posible a medida que actúa.

En este tipo de aprendizaje, el agente se enfrenta a una serie de situaciones (estados) y debe elegir una acción en cada una de ellas. Tras ejecutar una acción, el entorno responde con una nueva situación y una recompensa, que indica si la acción fue beneficiosa o no. A través de muchas interacciones, el agente va ajustando su comportamiento para tomar decisiones que lo acerquen a su objetivo.

Uno de los algoritmos más conocidos en este campo es Q-learning, que permite al agente aprender qué acciones tomar en cada estado mediante el uso de una tabla Q. Esta tabla guarda el valor esperado de realizar cada acción en cada estado y se actualiza en cada paso con la información que el agente va recogiendo.

Para decidir qué acción tomar, se usa la política e-greedy. Esto significa que, con una probabilidad epsilon, elige una acción al azar para seguir explorando, y con una probabilidad 1 - epsilon, elige la mejor acción, es decir, la que tiene el valor más alto en la tabla Q. Después de realizar la acción y recibir una recompensa, el agente actualiza la tabla Q usando la política greedy, ya que usa el mejor valor del siguiente estado, suponiendo que en el futuro el agente elegirá siempre la mejor acción posible.

En este proyecto se ha aplicado el algoritmo Q-learning a tres entornos propuestos por la librería Gymnasium: CliffWalking, Taxi y Blackjack. Los cuales se van a ver a continuación.

CLIFF WALKING

Explicación del entorno

El entorno CliffWalking-v0 consiste en una cuadricula de 4 filas por 12 columnas en la que un agente, en este caso el duende que realiza el recorrido, debe ir desde el punto de inicio, situado en la esquina inferior izquierda, hasta el objetivo, en la esquina inferior derecha. Entre ambos puntos, hay un acantilado que el agente debe evitar, ya que, si cae en él, recibe una gran penalización.

El espacio de observación está formado por 48 estados discretos, uno por cada celda de la cuadrícula. El espacio de acciones está formado por 4 acciones discretas, las cuales son: arriba (0), derecha (1), abajo (2) e izquierda (3). En cuanto a las recompensas, cada paso que da el agente penaliza con -1, pisar el acantilado penaliza con -100, y llegar al objetivo finaliza el episodio sin dar ningún tipo de recompensa.

Por lo tanto, el objetivo del agente es encontrar una ruta óptima que le permita alcanzar la meta evitando el acantilado y minimizando las penalizaciones.

Explicación Q table

A la hora de realizar este ejercicio, se ha generado una tabla Q la cual contiene los valores que el agente ha aprendido a lo largo del entrenamiento. Tiene 48 filas, una por cada estado del entorno, y 4 columnas, una por cada acción posible: arriba, derecha, abajo e izquierda. Al principio, todos los valores están inicializados a cero, pero a medida que el agente interactúa con el entorno y recibe recompensas o penalizaciones, los valores se actualizan siguiendo la fórmula de Q-learning. En cada paso, el agente elige una acción, observa la recompensa que obtiene y el nuevo estado al que llega, y con esa información ajusta el valor de la acción anterior. Para actualizar, tiene en cuenta la recompensa recibida y el mejor valor del nuevo estado, suponiendo que en el futuro seguirá eligiendo la mejor acción (política greedy). De esta forma, la tabla Q va aprendiendo qué acciones son mejores en cada estado según lo que el agente ha experimentado. Al final del entrenamiento, se pueden ver valores cercanos a −1 en los estados que forman parte del camino hacia la meta, y valores cercanos a −100 en las acciones que llevan al acantilado. También se observa que los estados del 37 al 47 tienen todos sus valores en cero, lo que significa que el agente nunca llegó a pasar por ellos, ya sea porque no eran necesarios para llegar al objetivo o porque el episodio se termina al alcanzarlo, como ocurre con el estado final.

Evolución del aprendizaje

Para mirar la evolución que ha tenido el agente hemos observado tres aspectos: la recompensa, los pasos que realizado hasta llegar al objetivo y si el agente ha obtenido éxito o no. Si miramos la gráfica de la recompensa podemos ver que la mayoría de las veces consigue una recompensa en torno a -13, la cual es la mejor recompensa que el agente puede conseguir en este entorno. Sin embargo, hay valores altos entorno al -100, que indican que le agente se ha caído al acantilado ya que el agente está aprendiendo. En cuanto a los pasos realizados por el agente, podemos ver similitudes con el de la recompensa ya que el valor óptimo de pasos realizados es 13, y la mayoría de los datos están en torno a ese número; también hay en algún episodio en el que da más pasos de la cuenta mientras el agente aprende. A la hora de observar los éxitos, vemos que al principio se van intercalando los éxitos(1) y los fracasos(0), pero a partir del episodio 1000 más o menos, se estabiliza en éxito, ya que el agente ha evolucionado adecuadamente; además también se puede observar en el porcentaje de éxito acumulado, en el cual vemos que la gráfica tiende hacia arriba llegando a obtener éxito a partir de los primeros episodios.

Elección de hiperparámetros y análisis de los resultados

Se ha realizado este mismo entrenamiento con otros hiperparámetros con el objetivo de ver con que combinación de hiperparámetros se obtiene una mayor recompensa. En este caso, se ha decidido combinar distintos valores de los hiperparámetros: learning_rate, gamma y decay_rate. Al observar la recompensa obtenida por cada combinación de hiperparámetros y su respectiva gráfica, podemos hacernos una idea de cual tiene mejor resultado. Destaca la combinación (0.7, 0.95, 0.001), la cual tiene un learning_rate que permite que el agente aprenda adecuadamente sin sobreajustar, un gamma que favorece el equilibrio entre recompensa a corto y largo plazo, y un decay_rate bajo que hace que la exploración dure más tiempo y que el agente pueda encontrar caminos más eficientes. Sin embargo, hay combinaciones que no han

sido tan ventajosas. La combinación(0.3, 0.95, 0.005) ha sido la peor de todas; esta combinación tiene un learning_rate bajo, que hace que el agente aprenda más lentamente, junto con un gamma alto, el cual da más importancia a las recompensas a largo plazo y un decay_rate más elevado, que no le ha permitido explorar lo suficiente. Por lo que, con esta combinación el agente obtiene caminos peores y con más caídas al acantilado.

TAXI

Explicación del entorno

El entorno Taxi-v3 consiste en una cuadrícula de 5 filas por 5 columnas en la que un agente, en este caso el taxi, debe recoger a un pasajero que se encuentra en una parada específica y llevarlo hasta su destino. Hay cuatro lugares que pueden ser tanto puntos de recogida como de entrega: rojo, verde, amarillo y azul. El taxi, el pasajero y el destino se colocan aleatoriamente al inicio del episodio.

El espacio de observación está formado por 500 estados discretos, que representan todas las posibles combinaciones entre la posición del taxi(25 casillas), la ubicación del pasajero (5 opciones, incluyendo cuando va dentro del taxi) y el destino (4 posibles ubicaciones).

El espacio de acciones está formado por 6 acciones discretas: moverse hacia el sur (0), norte (1), este (2), oeste (3), recoger al pasajero (4) y dejarlo en su destino (5).

En cuanto a las recompensas, cada paso que da el agente penaliza con -1, intentar recoger o dejar al pasajero de forma incorrecta penaliza con -10, y entregar correctamente al pasajero en su destino otorga una recompensa de 20.

Por lo tanto, el objetivo del agente es aprender a recoger al pasajero y llevarlo a su destino de la forma más eficiente posible, obteniendo la mayor recompensa posible.

Explicación Q table

La tabla Q generada en este ejercicio tiene 500 filas, una por cada estado del entorno, y 6 columnas, una por cada acción posible. Al inicio, todos los valores eran cero, pero tras el entrenamiento, se han ajustado según ha ido evolucionando el agente. En las primeras filas ya aparecen valores positivos, especialmente en acciones de movimiento o recogida, lo que indica que el agente ha empezado a identificar acciones útiles y va aprendiendo. En las últimas filas, algunos valores alcanzan cifras altas (como 16 o 18), lo que indica que esas acciones llevan a completar el ejercicio con éxito. También hay valores negativos, que corresponden a penalizaciones por acciones erróneas, como dejar al pasajero en un lugar incorrecto. Esto demuestra que el agente ha aprendido a elegir las acciones más adecuadas en función del estado en el que se encuentre.

Evolución del aprendizaje

Para analizar la evolución del agente en este entorno observamos los tres aspectos que habíamos mirado en el anterior: la recompensa, el número de pasos necesarios para completar la tarea y si el agente ha logrado el objetivo. En la gráfica de la recompensa podemos ver que los valores oscilan entre positivos y negativos, con muchas recompensas en torno a valores

cercanos a cero o superiores, lo que indica que el agente, en la mayoría de episodios, completa la tarea de forma eficiente, aunque todavía comete errores en algunos casos. En cuanto al número de pasos por episodio, se observa que en la mayoría de los casos el agente necesita entre 10 y 20 pasos, lo que refleja que ha aprendido un camino relativamente eficaz para recoger y entregar al pasajero. Respecto a los éxitos, se observa que, tras los primeros episodios, el agente comienza a resolver correctamente el ejercicio, estabilizándose en un éxito continuo. Esto también se aprecia claramente en la evolución del porcentaje de éxito acumulado, donde la curva asciende rápidamente hasta obtener un éxito completo, lo que confirma que el agente ha aprendido a completar la tarea de manera eficaz a lo largo de los episodios.

Elección de hiperparámetros y análisis de los resultados

Al igual que en el ejercicio anterior, se han probado distintas combinaciones de los hiperparámetros learning_rate, gamma y decay_rate para comprobar cuál proporciona mejores resultados. Al analizar la recompensa media obtenida en los últimos 100 episodios y su gráfica correspondiente, se puede ver qué configuraciones han sido más efectivas. La combinación (0.7, 0.8, 0.05) ha sido la mejor de todas, la cual tiene la mejor recompensa media. Esta combinación permite que el agente aprenda con rapidez sin que sobreajuste, se centre en recompensas más inmediatas, lo cual es útil en entornos como Taxi donde los objetivos se logran en pocos pasos, y reduce la exploración con un decay_rate de 0.05 que favorece la explotación de lo aprendido. En cambio, la combinación (0.7, 0.95, 0.05) ha sido la menos eficiente. Aunque mantiene el learning_rate es el mismo(0.7), el gamma alto hace que el agente valore más la recompensa a largo plazo, y al unirse con un decay_rate también alto, la exploración se reduce demasiado pronto. Esto puede haber provocado que el agente no explorara lo suficiente al inicio y no adoptara los mejores caminos.

BLACJACK (VOLUNTARIO)

Explicación del entorno

El entorno Blackjack-v1 es un juego en el que un agente, el jugador, debe decidir si pedir una carta más (hit) o plantarse (stick), con el objetivo de conseguir una suma de cartas lo más cercana posible a 21 sin pasarse. El dealer también.

El espacio de observación está formado por una tupla de tres elementos: la suma actual de las cartas del jugador (entre 4 y 21), el valor de la carta del dealer (entre 1 y 10), y si el jugador tiene un as (0 o 1). Para realizar el código de manera más sencilla, se ha transformado esta tupla a un diccionario que asigna un número único a cada combinación posible. El espacio de acciones es discreto y está formado por 2 posibles acciones: plantarse (0) o pedir carta (1). En cuanto a las recompensas, si el jugador gana la partida recibe 1, si pierde −1, y si hay empate 0.

El objetivo del agente es aprender a tomar la mejor decisión en cada estado (pedir o plantarse), intentando ganar, pero sin pasarse de 21, y adaptándose al valor visible de la carta del dealer.

Explicación Q table

La tabla Q generada en este ejercicio contiene una fila por cada estado posible y 2 columnas, correspondientes a las dos acciones disponibles en el entorno, plantarse (0) o pedir carta (1).

Al comenzar el entrenamiento, todos los valores eran cero, pero a medida que el agente interactúa con el entorno, los valores se van ajustando en función de las recompensas obtenidas. En la tabla resultante pueden observarse tanto valores positivos como negativos. Los positivos indican que esas acciones han llevado a ganar una partida, mientras que los negativos reflejan acciones que suelen llevar a perder. También hay estados con valores cercanos a cero, especialmente aquellos que el agente ha visitado. Esto indica que el agente ha logrado identificar qué decisiones le acercan a ganar y cuáles no.

Evolución del aprendizaje

Para analizar la evolución del agente en el entorno de Blackjack, se han tenido en cuenta tres la recompensa, el número de cartas repartidas y los resultados obtenidos (ganar, perder o empatar). En la gráfica de recompensas, se observa que los valores varían entre -1, 0 y 1(perder, empatar y ganar). Esto indica que el agente, aunque consigue algunas victorias, también comete errores. En cuanto al número de cartas repartidas, la mayoría de los episodios se resuelven en pocas jugadas, normalmente entre una y tres cartas, lo cual es esperable en este juego donde se debe decidir rápidamente si plantarse o seguir pidiendo. Por último, al observar los promedios obtenidos, vemos que el porcentaje más alto es el de las derrotas(57%), lo que indica que el agente ha aprendido a jugar, pero aún comete errores que lo llevan a perder con frecuencia.

Elección de hiperparámetros y análisis de los resultados

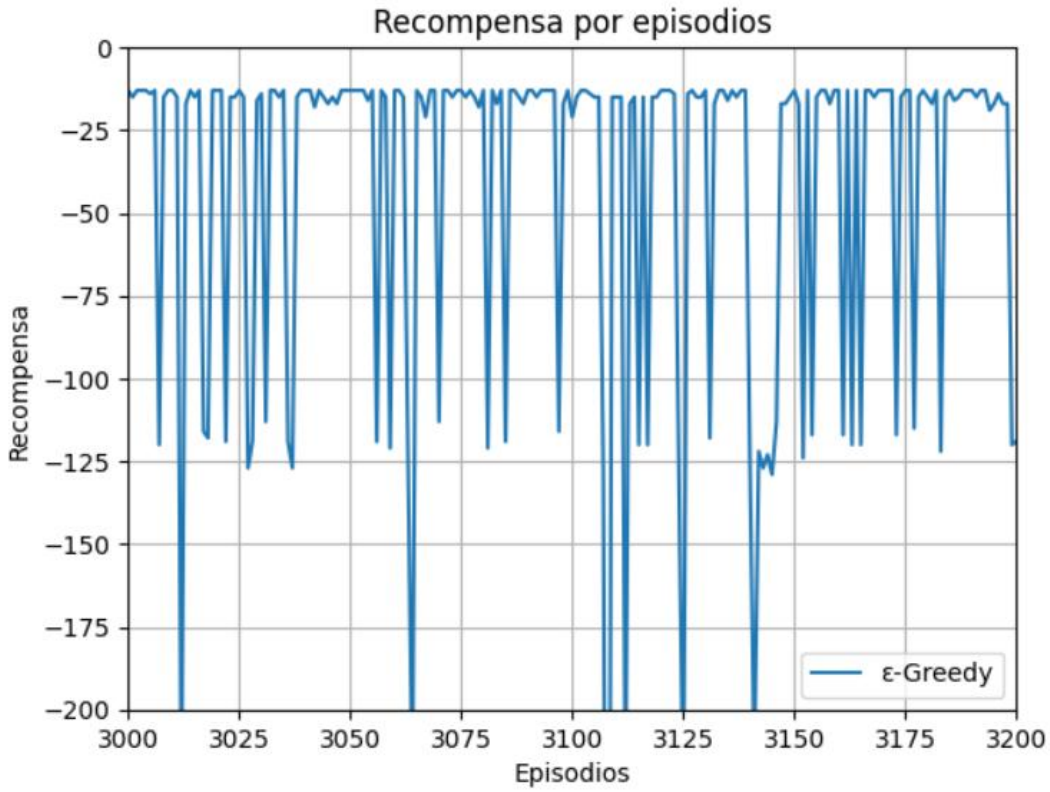
Al igual que en los ejercicios anteriores, se han probado distintas combinaciones de hiperparámetros para ver cuál proporciona mejores resultados. La combinación (0.1, 0.8, 0.01) ha sido la más efectiva. El learning_rate bajo permite un aprendizaje más estable, lo cual es útil en Blackjack donde las cartas se reparten de forma aleatoria y los resultados pueden variar mucho entre episodios. Esto evita que el agente se vea demasiado influido por partidas puntuales. El gamma de 0.8 hace que el agente priorice recompensas inmediatas, algo adecuado en este entorno. Además, el decay_rate bajo hace que le agente explore y así descubrir mejores estrategias. En cambio, la combinación (0.9, 0.95, 0.01) ha sido la menos eficiente. Aunque el decay_rate es el mismo, el learning_rate alto hace que el agente aprenda de partidas concretas, de situaciones puntuales en un entorno muy aleatorio como es el Blackjack . Además, un gamma alto implica que el agente dé demasiada importancia a las recompensas a largo plazo, algo que en partidas cortas y en un entorno tan aleatorio no es lo mejor.

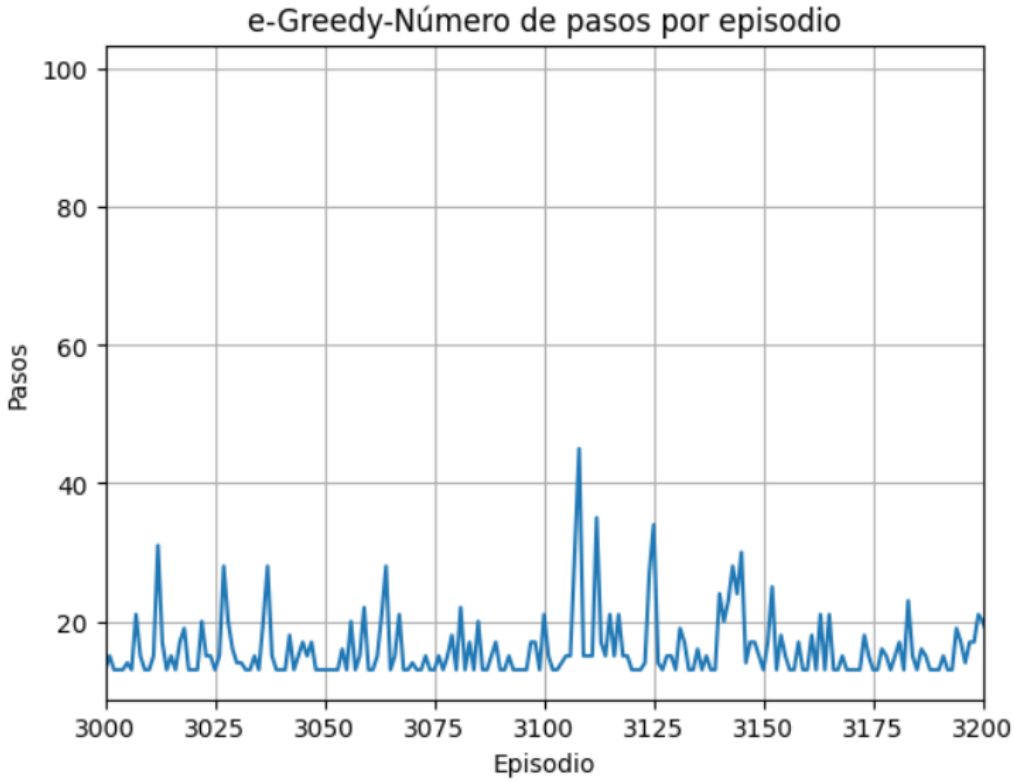
CONCLUSIÓN

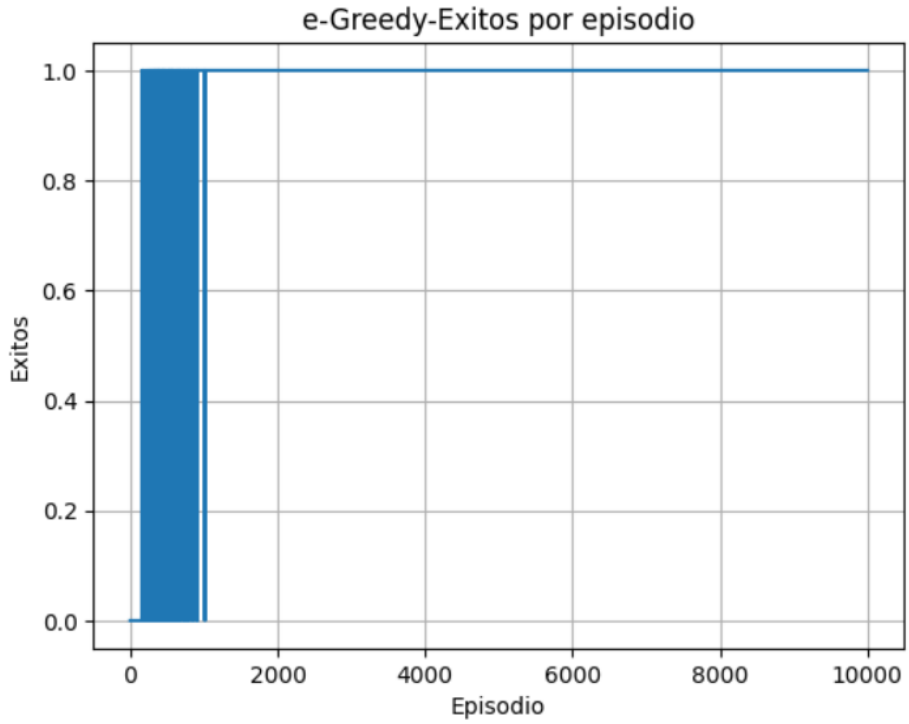
A lo largo de este proyecto se ha implementado el algoritmo Q-learning en tres entornos distintos: CliffWalking, Taxi y Blackjack, cada uno de manera diferente. Se ha demostrado que el agente es capaz de aprender a través de la interacción con el entorno, ajustando su tabla Q según las recompensas recibidas. También se ha analizado el impacto de los hiperparámetros en el rendimiento, comprobando que, si se ajustan correctamente según el entorno, el aprendizaje por refuerzo permite al agente mejorar su comportamiento, evitar errores y adoptar estrategias más eficientes.

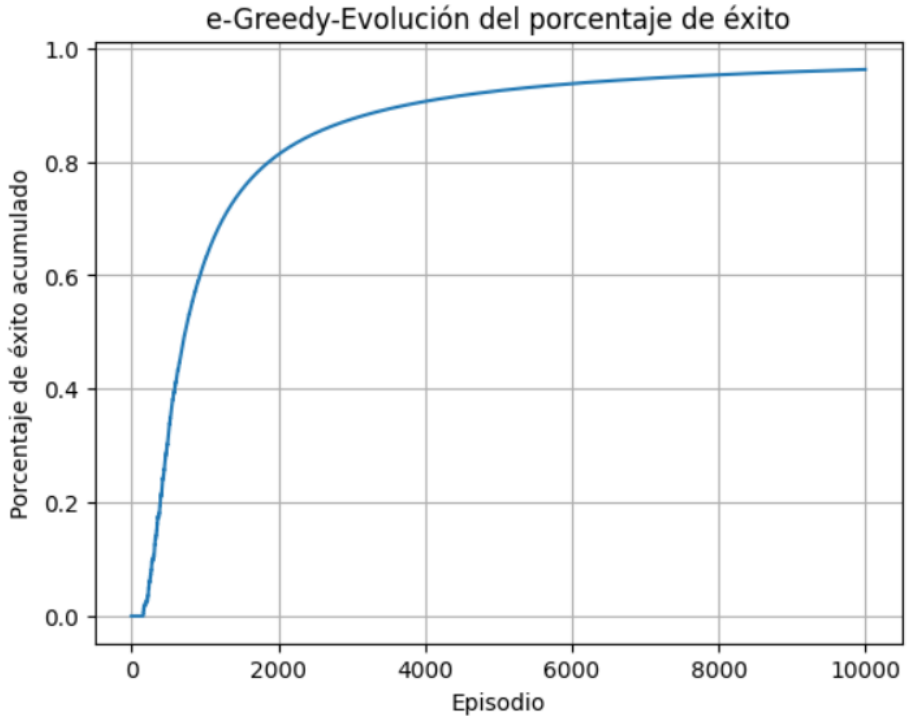
ANEXO

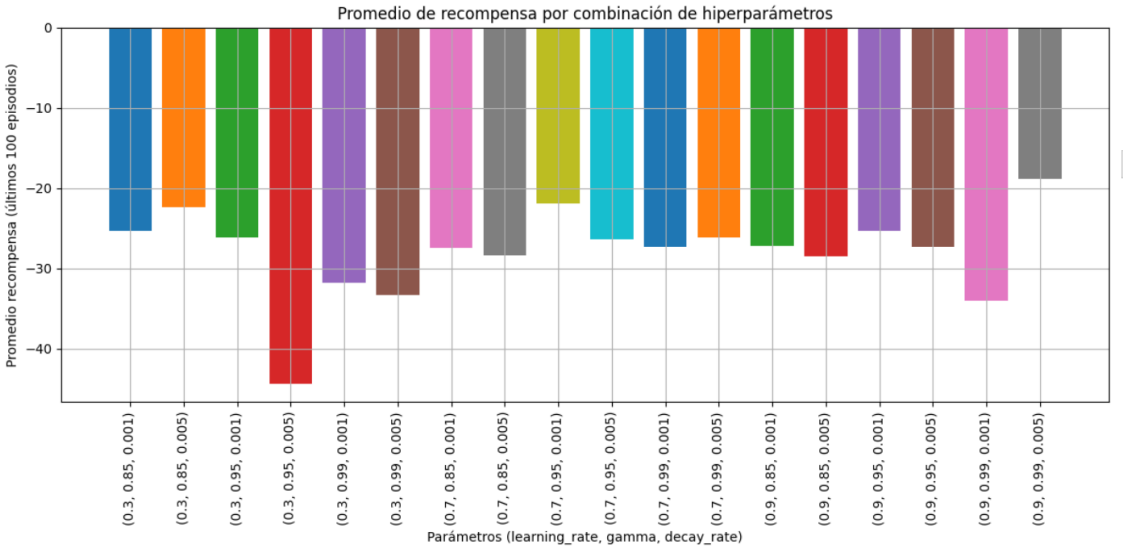
Imágenes Cliff



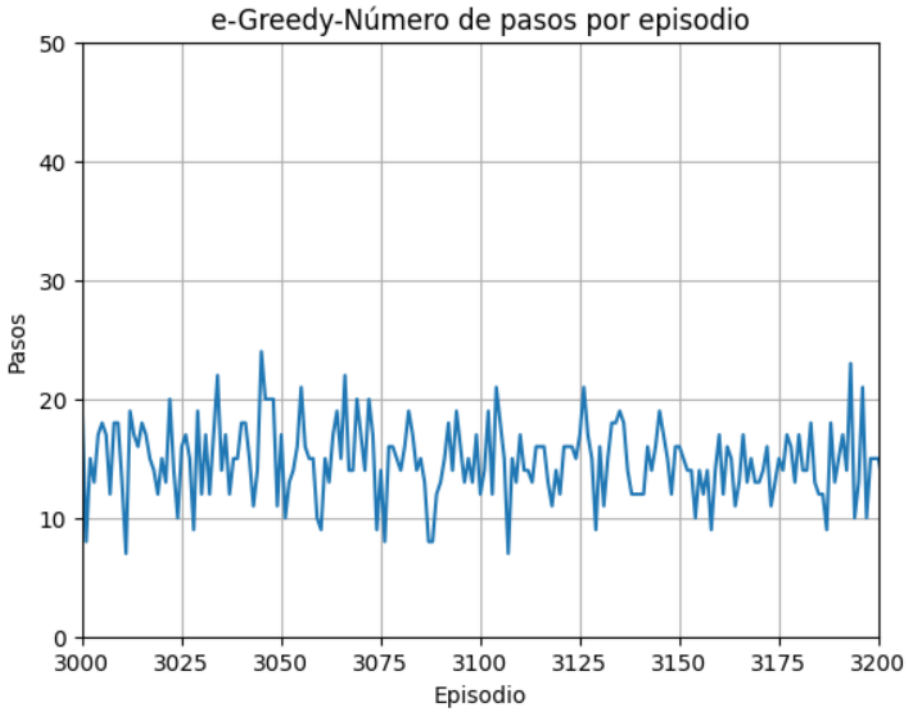


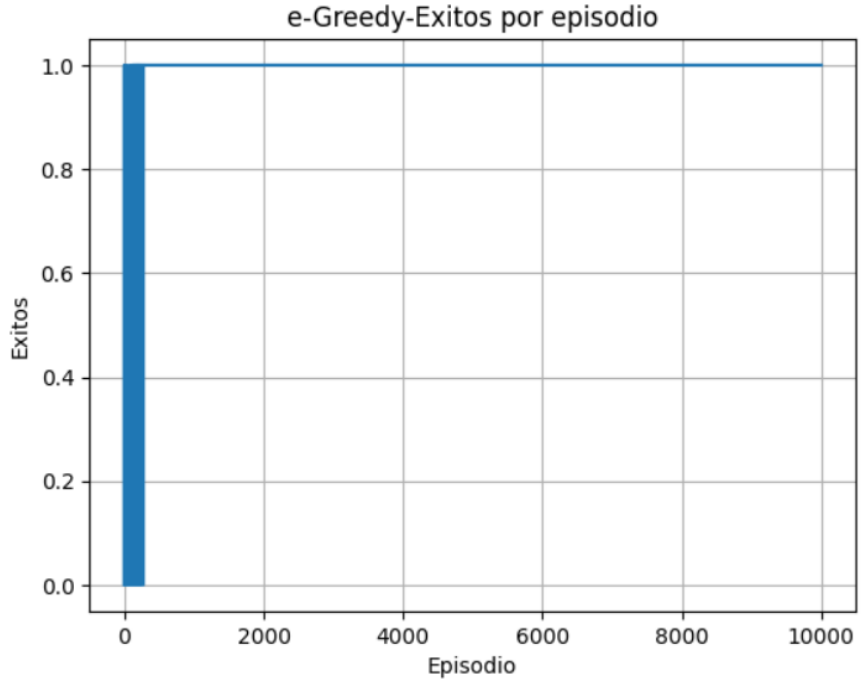

e-Greedy-Exitos por episodio
Exitos
Episodio
0.0
0.2
0.4
0.6
0.8
1.0
0
2000
4000
6000
8000
10000

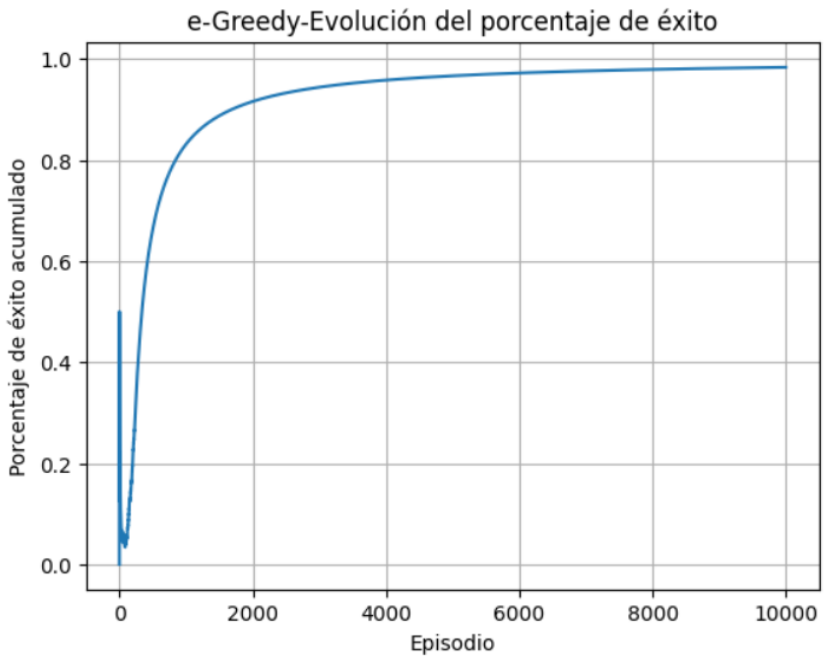

e-Greedy-Evolución del porcentaje de éxito
Porcentaje de éxito acumulado
Episodio
0.0
0.2
0.4
0.6
0.8
1.0
0
2000
4000
6000
8000
10000

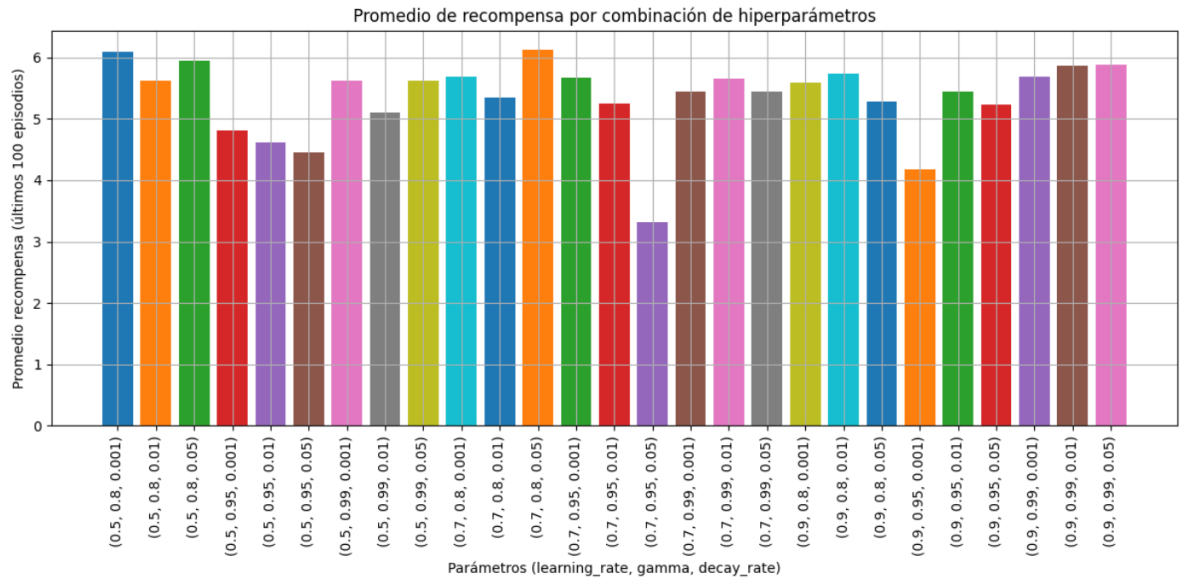


Imágenes Taxi

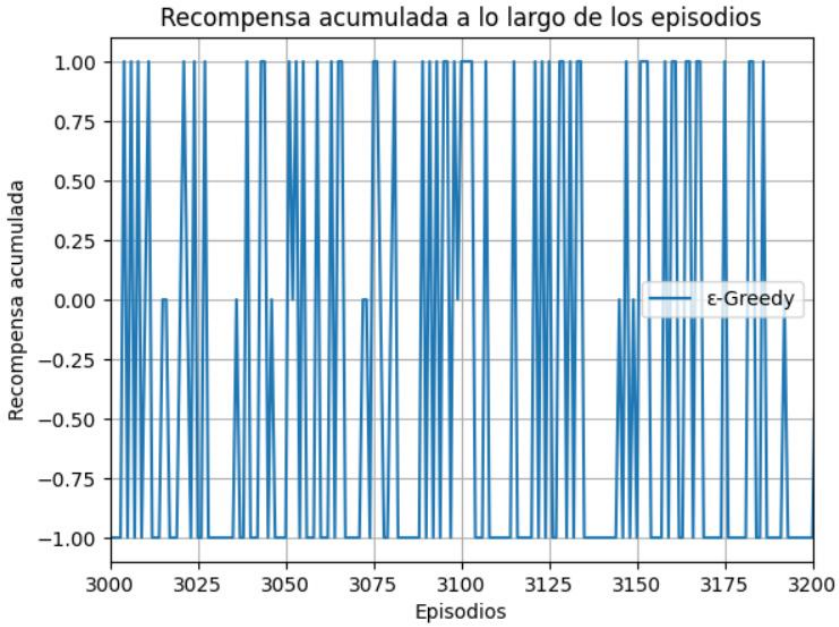

e-Greedy-Número de pasos por episodio
Pasos
Episodio
0
10
20
30
40
50
3000
3025
3050
3075
3100
3125
3150
3175
3200


e-Greedy-Exitos por episodio
Exitos
Episodio
0.0
0.2
0.4
0.6
0.8
1.0
0
2000
4000
6000
8000
10000





Imágenes Blackjack




```
print(f"Promedio de exitos: {exitos_e_greedy_bj/episodios}")
print(f"Promedio de perdidas: {perdidas_e_greedy_bj/episodios}")
print(f"Promedio de empates: {empates_e_greedy_bj/episodios}")
```

```
Promedio de exitos: 0.3607
Promedio de perdidas: 0.5704
Promedio de empates: 0.0689
```